

## 1. Обґрунтування вибору методу

Нерп: 43	Несп: 4	Go !
----------	---------	------

  

$A * x = b:$	<input type="text" value="6.7"/>	<input type="text" value="1.16"/>	<input type="text" value="0.91"/>	<input type="text" value="1.18"/>	<input type="text" value="-0.36"/>	$* x =$	<input type="text" value="2.1"/>
	<input type="text" value="1.04"/>	<input type="text" value="3.72"/>	<input type="text" value="1.3"/>	<input type="text" value="-1.63"/>	<input type="text" value="0.12"/>		<input type="text" value="0.48"/>
	<input type="text" value="1.03"/>	<input type="text" value="-2.46"/>	<input type="text" value="5.88"/>	<input type="text" value="2.1"/>	<input type="text" value="0.583"/>		<input type="text" value="1.29"/>
	<input type="text" value="1.3"/>	<input type="text" value="0.16"/>	<input type="text" value="2.1"/>	<input type="text" value="5.66"/>	<input type="text" value="-6"/>		<input type="text" value="6.44"/>
	<input type="text" value="0.84"/>	<input type="text" value="-0.78"/>	<input type="text" value="-0.317"/>	<input type="text" value="3"/>	<input type="text" value="1"/>		<input type="text" value="-0.48"/>

  

Примітка:	<input type="text" value="система є несиметричною"/>
-----------	--

Оскільки система є несиметричною, доцільно реалізовувати метод Гаусса.

## 2. Лістинг

```
#include<iostream>
#include<cmath>
#include<iomanip>
using namespace std;

int mul_func(float **mat1, float **mat2, float **res_mat, int m){
    int i, j, k;

    for(i = 0; i < m; i++){
        for(j = 0; j < m; j++){
            res_mat[i][j] = 0;
            for(k = 0; k < m; k++){
                res_mat[i][j]=res_mat[i][j]+mat1[i][k]*mat2[k][j];
            }
        }
    }

    return 0;
}

int gau_fun(float **mat, float *res, float &det, int m) {
    int i, j, k, max_index;
    float maxie;
    float matr[m][m + 1];

    for(i = 0; i < m; i++)
        for(j = 0; j < m + 1; j++)
            matr[i][j] = mat[i][j];
```

```

for(k = 0; k < m; k++) {

    maxie = matr[k][k];

    for(i = k; i < m; i++) {
        if (matr[i][k] > maxie) {maxie = matr[i][k]; max_index = i;}
    }

    if(maxie == 0) return 0;

    for(int l = k + 1; l < m; l++) {
        float koef = matr[l][k]/matr[k][k];
        for(j = 0; j < m + 1; j++)
            matr[l][j] = matr[l][j] - matr[k][j]*koef;
    }
}

res[m - 1] = matr[m - 1][m]/matr[m - 1][m - 1];

for(i = m - 2; i >= 0; i--) {
    res[i] = matr[i][m]/matr[i][i];
    for(j = i + 1; j < m; j++) res[i] = res[i] - matr[i][j]/matr[i][i]*res[j];
}

det = 1;
for(i = 0; i < m; i++)
    for(j = i; j < i + 1; j++)
        det = det*matr[i][j];

return 1;
}

int main() {

    int i, j, k, l, n;
    float deter;

    cout << "Input number of unknown variables: ";
    do {
        k = scanf("%i", &n);
        fflush(stdin);
        if (k != 1) cout << "Error, input value again: ";
    } while(k != 1);

    float **matrix = new float*[n];
    for(i = 0; i < n; i++)
        matrix[i] = new float[n + 1];

    float *result = new float[n];

    float **matrix_reverse = new float*[n];
    for(i = 0; i < n; i++)

```

```

        matrix_reverse[i] = new float[n];

float **matrix_mult = new float*[n];
for(i = 0; i < n; i++)
    matrix_mult[i] = new float[n];

cout << "\nInput matrix A: \n";

for(i = 0; i < n; i++)
    for(j = 0; j < n + 1; j++){
        cout << "Input a[" << i+1 << ", " << j+1 << "]: ";
        do {
            k = scanf("%f", &matrix[i][j]);
            fflush(stdin);
            if (k != 1) cout << "Error, input value again: ";
        } while(k != 1);
    }

cout << "\nMatrix A: \n";

for(i = 0; i < n; i++){
    for(j = 0; j < n + 1; j++)
        cout << matrix[i][j] << " ";
    cout << "\n";
}

if(gau_fun(matrix, result, deter, n) == 0) {cout << "\nError!\n";}
else {
    cout << "\nRoots obtained: \n";
    for(i = 0; i < n; i++){
        cout << result[i] << " ";
    }

    cout << "\n\ndet A = " << deter << "\n";

    for(k = 0; k < n; k++){
        matrix[k][n] = 1;
        for(l = 0; l < n; l++){
            if(l != k) matrix[l][n] = 0;
        }

        gau_fun(matrix, result, deter, n);

        for(i = 0; i < n; i++)
            matrix_reverse[i][k] = result[i];
    }

    cout << "\nReverse matrix: \n";

    for(i = 0; i < n; i++){
        for(j = 0; j < n; j++)
            cout << matrix_reverse[i][j] << " ";
        cout << "\n";
    }
}

```

```

mul_func(matrix, matrix_reverse, matrix_mult, n);

    cout << "\nA*A^(-1): \n";

    cout.setf(ios_base::fixed);
    cout.precision(10);
    for(i = 0; i < n; i++){
        for(j = 0; j < n; j++){
            cout << matrix_mult[i][j] << " ";
            cout << "\n";
        }

        cout << "\n";
        float *b_res = new float[n];
        for (i=0; i<n; i++)
            b_res[i]=matrix[i][0]*result[0] +
                matrix[i][1]*result[1]+ matrix[i][2]*result[2]+
                matrix[i][3]*result[3]+ matrix[i][4]*result[4];

        for(i=0; i<n; i++)
            cout << "\n"<<matrix[i][n]-b_res[i];
    }

    return 0;
}

```

### 3. Результати роботи програми

```
C:\Users\Sashko\Desktop\IASA\4 semester\Numerical analysis (1st)\Lab2 (23.03.16)\Lab2.exe
Input matrix A:
Input a[1, 1]: 6.7
Input a[1, 2]: 1.16
Input a[1, 3]: 0.91
Input a[1, 4]: 1.18
Input a[1, 5]: -0.36
Input a[1, 6]: 2.1
Input a[2, 1]: 1.04
Input a[2, 2]: 3.72
Input a[2, 3]: 1.3
Input a[2, 4]: -1.63
Input a[2, 5]: 0.12
Input a[2, 6]: 0.48
Input a[3, 1]: 1.03
Input a[3, 2]: -2.46
Input a[3, 3]: 5.88
Input a[3, 4]: 2.1
Input a[3, 5]: 0.583
Input a[3, 6]: 1.29
Input a[4, 1]: 1.3
Input a[4, 2]: 0.16
Input a[4, 3]: 2.1
Input a[4, 4]: 5.66
Input a[4, 5]: -6
Input a[4, 6]: 6.44
Input a[5, 1]: 0.84
Input a[5, 2]: -0.78
Input a[5, 3]: -0.317
Input a[5, 4]: 3
Input a[5, 5]: 1
Input a[5, 6]: -0.48

Matrix A:
6.7 1.16 0.91 1.18 -0.36 2.1
1.04 3.72 1.3 -1.63 0.12 0.48
1.03 -2.46 5.88 2.1 0.583 1.29
1.3 0.16 2.1 5.66 -6 6.44
0.84 -0.78 -0.317 3 1 -0.48

Roots obtained:
0.205991 0.0556653 0.252892 0.103785 -0.840802

det A = 3208.68

Reverse matrix:
0.172831 -0.0697021 -0.00657538 -0.0222738 -0.0592258
-0.0613591 0.283373 -0.0554629 0.0305178 0.159348
-0.0323412 0.0957356 0.147716 0.00894998 -0.0555496
-0.0577637 0.0720118 -0.0100698 0.0539892 0.300369
-0.0299993 0.0938931 0.0392977 -0.116617 0.255323

A*A^(-1):
1.00000000000 0.00000000596 0.0000000047 -0.0000000075 -0.00000000149
0.00000000247 1.00000001192 0.0000000005 -0.0000000019 0.00000000224
0.00000000298 -0.00000001378 1.00000000000 -0.00000000149 -0.00000000298
0.00000000149 0.00000000000 0.00000000000 1.00000001192 0.00000001192
0.00000000056 -0.00000000298 0.0000000112 0.00000000075 1.00000000000
```